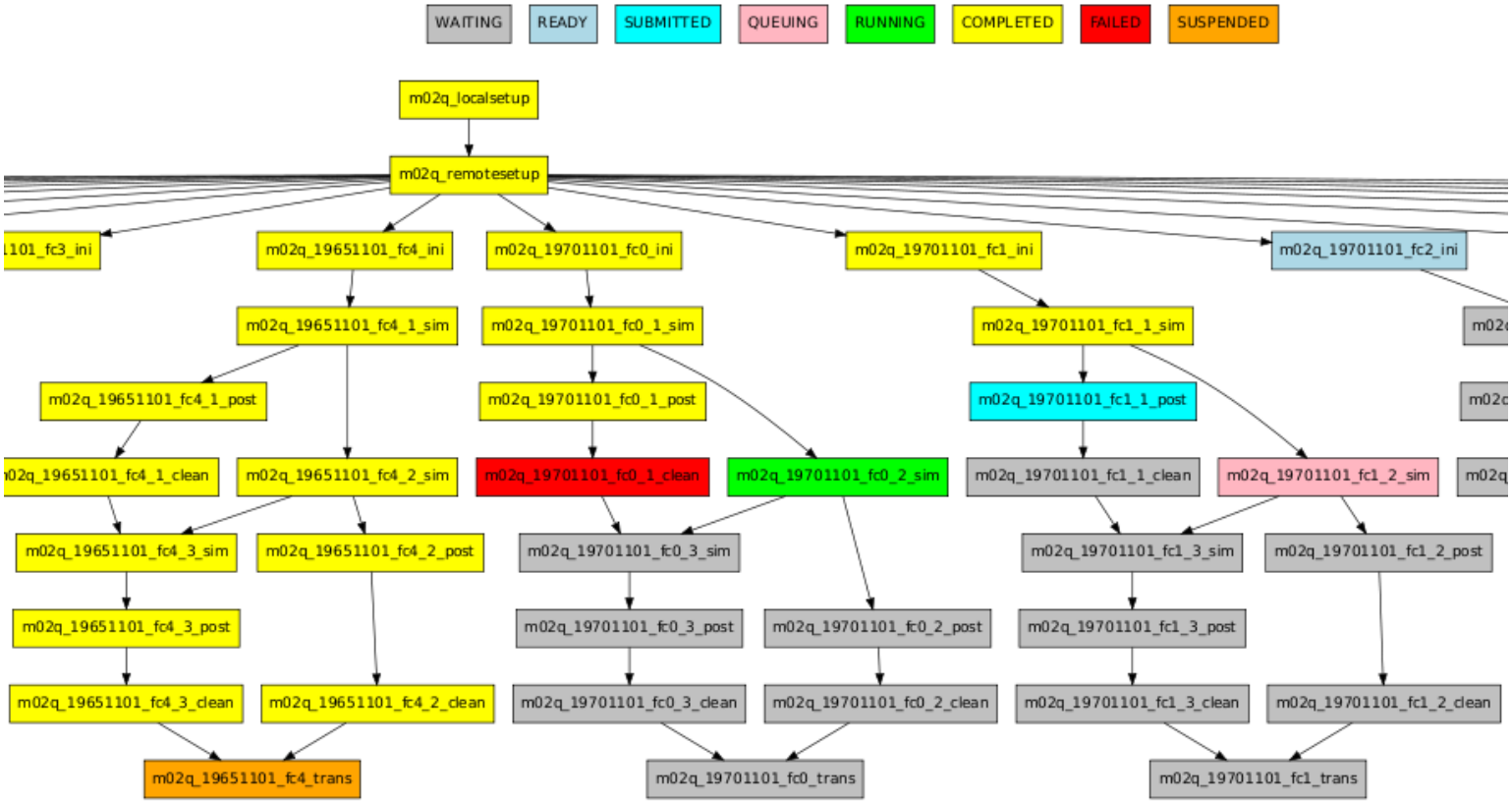
Autosubmit 3.0.0



M. Asif, D. Manubens, O. Mula

Outline

- Autosubmit Documentation
  - Wiki
  - Installation
- Compatibility remark
- Typical Experiment
  - Workflow
  - Autosubmit steps
- Git Project (A model)
  - GitLab
  - Experiment Definition File
  - Git Submodules
- Develop & Test
- To Do List ...

Wiki

http://ic3.cat/wikicfu/index.php/Tools/Autosubmit

Autosubmit

Autosubmit launches and monitors experiments on any platform used at CFU. A general description of what is a typical climate forecast experiment and what is the goal of Autosubmit, more technical description of the architecture and how it works, how to install on your computer, user's manual and documentation and Autosubmit developers' page is available here.

- Objective
- Description
- Requirements
- Use
- Repository
- Contact
- Development
- Style Guide
- List of experiments (only accessible form inside CFU network)
- SCRUM Framework

Installation

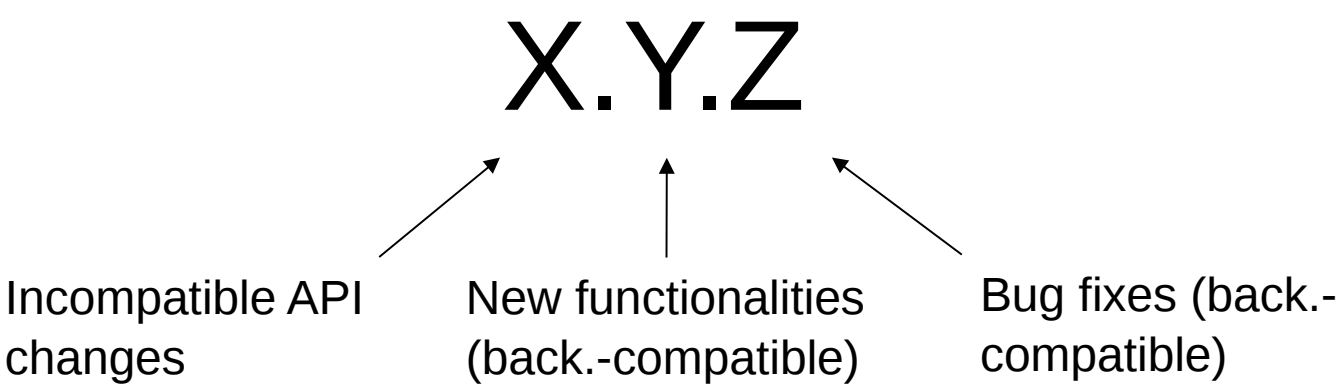
Python Package Index (PyPI)

Download
autosubmit-3.0.0a21.tar.gz

```
- Pre-requisties: These packages (bash, python2, python-argparse, python-dateutil,
python-pydot, python-matplotlib, sqlite3, git-scm > 1.8.2) must be available at local machine.
And the machine is also able to access HPC's/Clusters via password-less ssh.

- Install Autosubmit
   > pip install autosubmit
  or download, unpack and "python setup.py install"

- Create a repository for experiments: Say for example "/cfu/autosubmit" then
  edit the repository path (LOCAL_ROOT_DIR) into autosubmit/config/dir_config.py

- Create a blank database: Say for example "autosubmit.db" at above created repository:
   > cp autosubmit/database/data/autosubmit.sql /cfu/autosubmit/
   > cd /cfu/autosubmit
   > sqlite3 autosubmit.db
   sqlite3>.read autosubmit.sql
   > chmod 775 autosubmit.db
  then edit the database file path and name (DB_DIR, DB_FILE, DB_NAME) into autosubmit/config/dir_config.py
```

Releasing

Semantic Versioning (http://semver.org)



Compatibility

- Running 2.x experiments with new version 3.x:
  - Copy experiment:

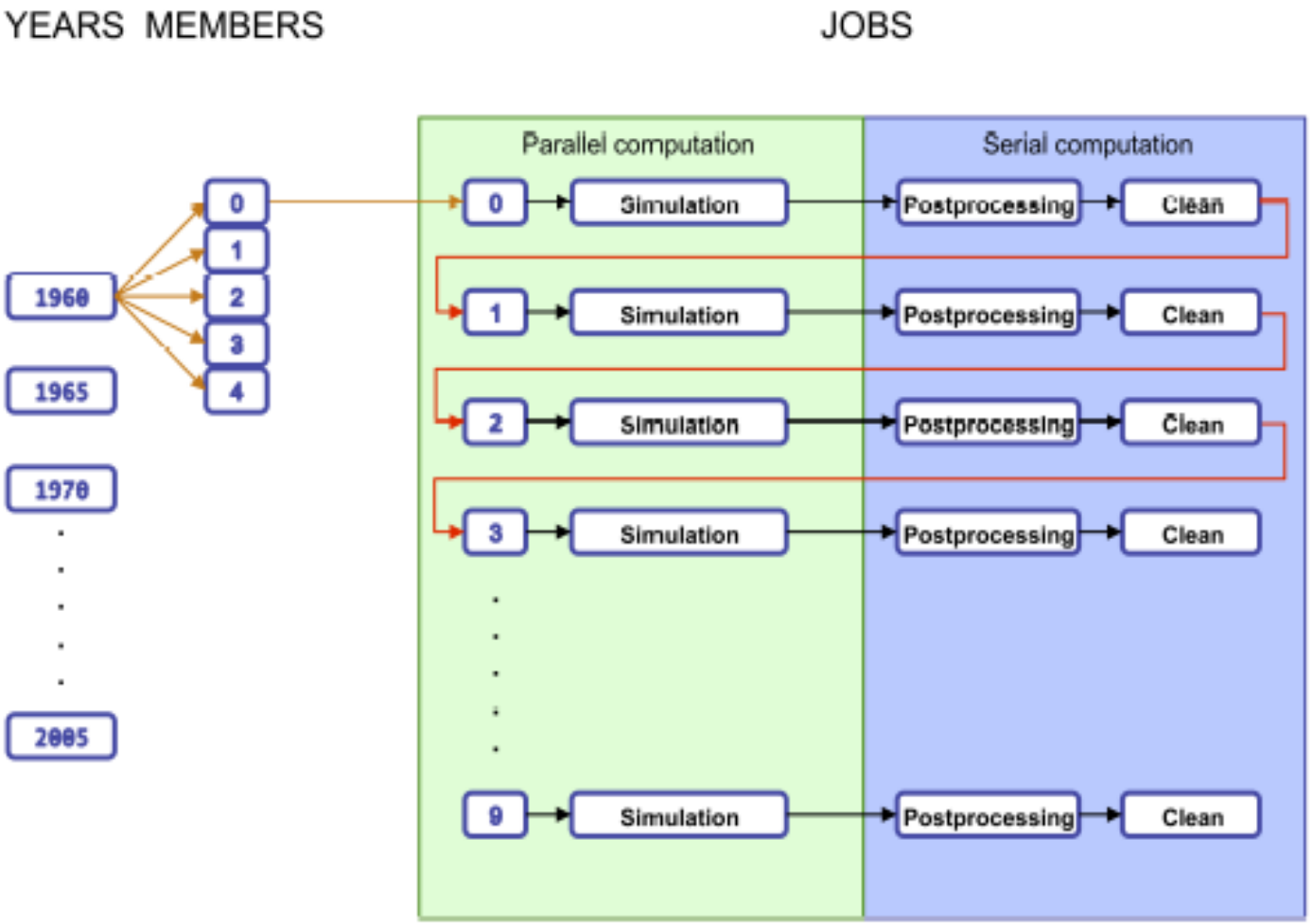
    > expid.py --copy m003 --HPC marenostrum3 \
    --description “Copy of m003”

  - Script to migrate Autosubmit 2.x experiments to Autosubmit 3.x

    > migrate_exp.sh *<ORIGINAL_EXPID>* *<EXPID>*

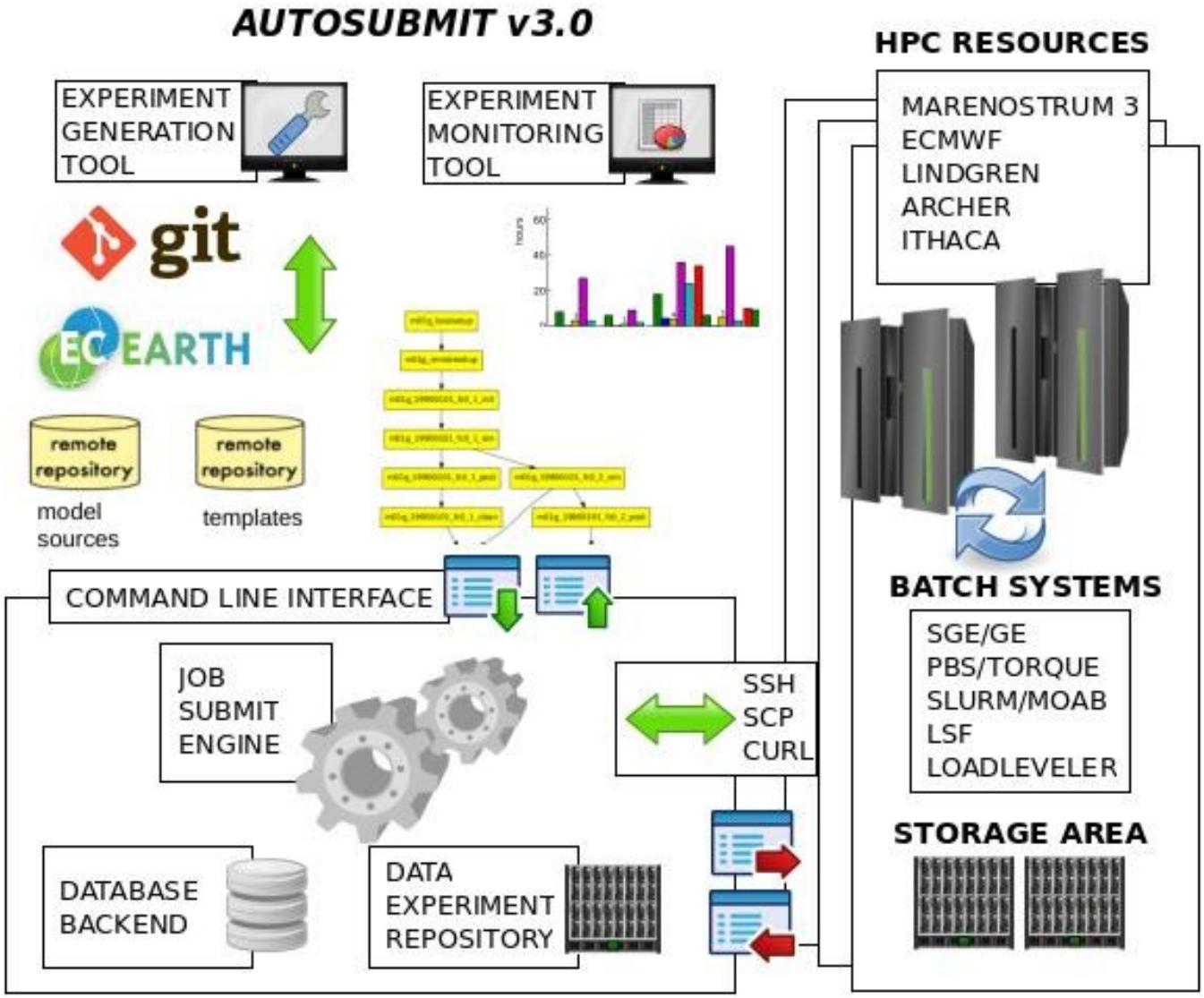
- Running 3.x experiments with old version 2.x: *not allowed !*

Typical Experiment



Workflow

- Choose platform that is available for performing experiments, and check if enough computing time is available.
- Check if the targeted model version has already been deployed.
- Check if the SSH access to the platform has been set appropriately.
- Autosubmit:
  - Multiple runs producing raw GRIB/NEMO output.
  - Postprocessing, converting raw output to standarized netCDF and computing some diagnostics.
  - Transfer the post-processed data to local server.
- Use the R package s2dverification, if necessary, updating the Load.R to reflect new data sets and new variables (diagnostics) available on the file system.


AUTOSUBMIT v3.0
EXPERIMENT GENERATION TOOL
EXPERIMENT MONITORING TOOL
git
EC EARTH
remote repository
model sources
remote repository
templates
COMMAND LINE INTERFACE
JOB SUBMIT ENGINE
SSH SCP CURL
DATABASE BACKEND
DATA EXPERIMENT REPOSITORY
HPC RESOURCES
MARENOSTRUM 3
ECMWF
LINDGREN
ARCHER
ITHACA
BATCH SYSTEMS
SGE/GE
PBS/TORQUE
SLURM/MOAB
LSF
LOADLEVELER
STORAGE AREA
enterprise:8000/autosubmit_v3

Autosubmit Steps

- Register experiment into database, create data experiment directory:

```
python expid.py -h
```

- Fill in configuration files

- Create experiment dependency tree, download GIT project:

```
python create_exp.py -h
```

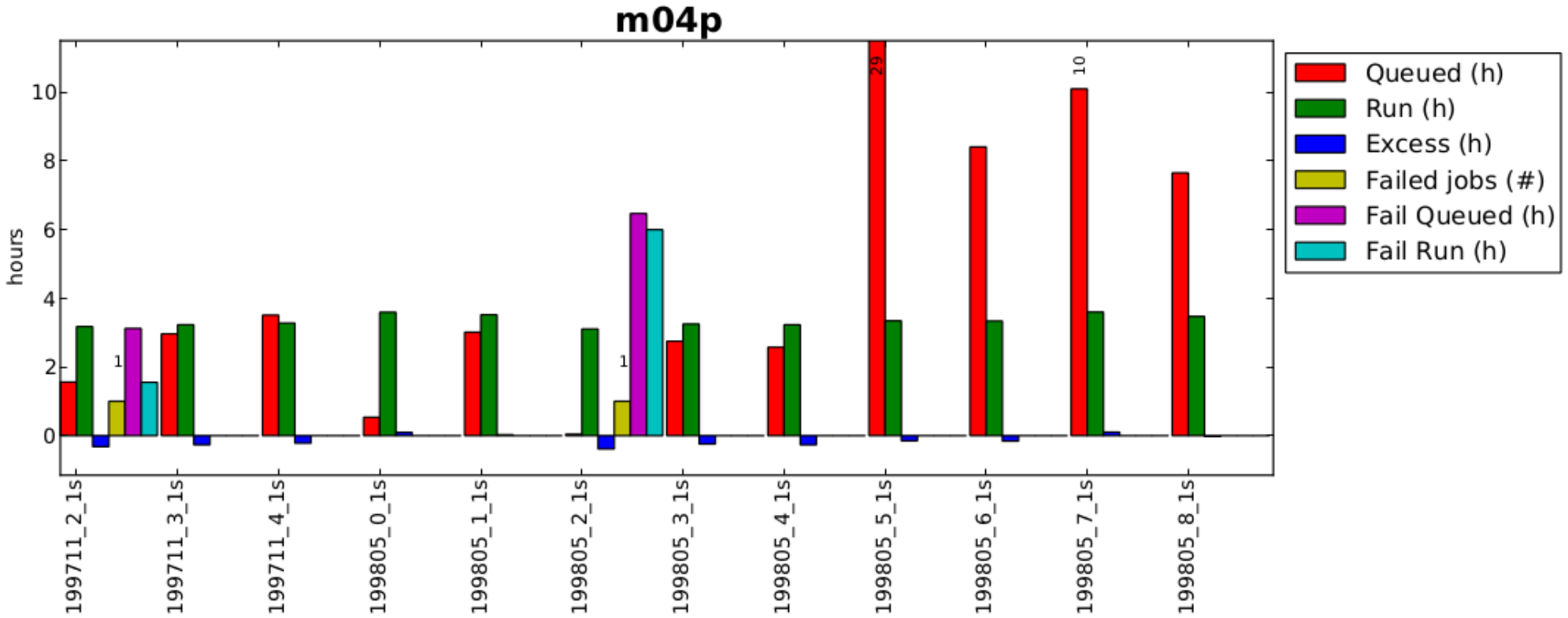
- Fill in parameters files

- Launch Autosubmit experiment:

```
python autosubmit.py -h
```

Monitor statistics

```
python statistics.py -h
```



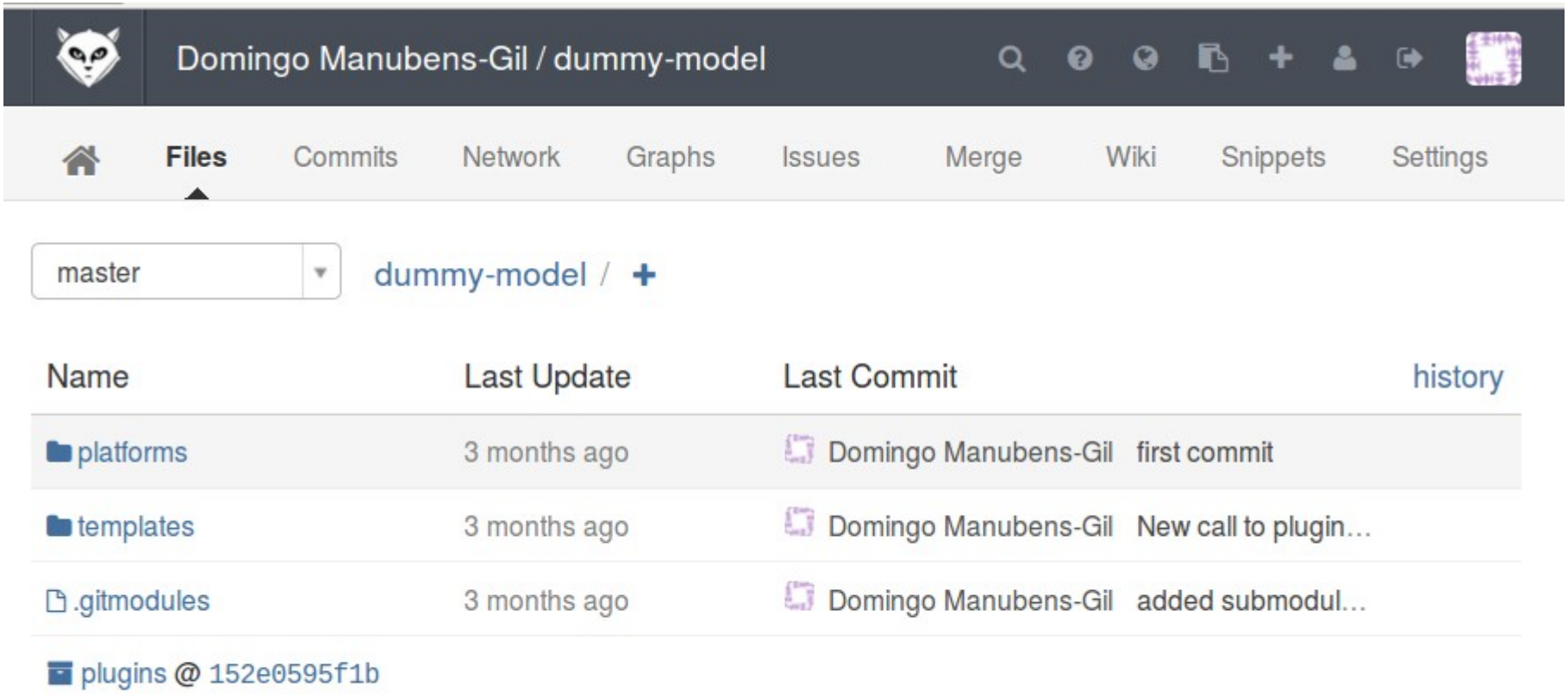
GIT Project

- Dummy model: sample project to begin
- Consists of:
  - Model sources (FORTRAN, C, …).
  - Templates (portable shell scripts) for 7 Autosubmit job types:

<table>
<tr><th>LOCAL</th><th colspan="3">REMOTE HPC</th><th>LOCAL</th></tr>
<tr><td rowspan="2">SETUP</td><td rowspan="2">SETUP</td><td>INI</td><td>SIM</td><td rowspan="2">TRANS</td></tr>
<tr><td>POST</td><td>CLEAN</td></tr>
</table>

  - Project parametrization file (key-value pairs) with default values
  - Platform specific parametrization file.
  - Plugins (additional shell script functions) common to all templates (GIT submodules) i.e. ocean diagnostics.

GitLab


Domingo Manubens-Gil / dummy-model
Files
Commits
Network
Graphs
Issues
Merge
Wiki
Snippets
Settings
master
dummy-model /
Name
Last Update
Last Commit
history
platforms
3 months ago
Domingo Manubens-Gil
first commit
templates
3 months ago
Domingo Manubens-Gil
New call to plugin…
.gitmodules
3 months ago
Domingo Manubens-Gil
added submodul…
plugins @ 152e0595f1b

Experiment Definition File

```
[git]
# type = str, help = 'https://gitlab.cfu.local/cfu/'
GIT_PROJECT_ORIGIN = https://gitlab.cfu.local/dmanubens/dummy-model.git
# type = str, default = 'master', help = {'master' (defualt), 'develop', ...}
GIT_PROJECT_BRANCH = master
# type = str, default = leave empty, help = if model branch is a TAG leave empty.
GIT_PROJECT_COMMIT =
# Where is HPC PLATFORM CONFIGURATION file location relative to GIT_PROJECT_NAME root path
GIT_FILE_PLATFORM_CONF = dummy-model/platforms/ithaca.conf
# Where is LOCALSETUP script location relative to GIT_PROJECT_NAME root path
GIT_FILE_LOCALSETUP = dummy-model/templates/common/common.localsetup
# Where is REMOTESETUP script location relative to GIT_PROJECT_NAME root path
GIT_FILE_REMOTESETUP = dummy-model/templates/common/common.remotesetup
# Where is TRANS script location relative to GIT_PROJECT_NAME root path
GIT_FILE_TRANS = dummy-model/templates/common/common.localtrans
# Where is MODEL CONFIGURATION file location relative to GIT_PROJECT_NAME root path
GIT_FILE_MODEL_CONF = dummy-model/templates/dummy1/dummy1.conf
# Where is INI script location relative to GIT_PROJECT_NAME root path
GIT_FILE_INI = dummy-model/templates/dummy1/dummy1.ini
# Where is SIM script location relative to GIT_PROJECT_NAME root path
GIT_FILE_SIM = dummy-model/templates/dummy1/dummy1.sim
# Where is POST script location relative to GIT_PROJECT_NAME root path
GIT_FILE_POST = dummy-model/templates/dummy1/dummy1.post
# Where is CLEAN script location relative to GIT_PROJECT_NAME root path
GIT_FILE_CLEAN = dummy-model/templates/dummy1/dummy1.clean
```

GIT Submodules

- How to add a submodule:

  > *git submodule add https://gitlab.cfu.local/cfu/dummy-plugins.git plugins*

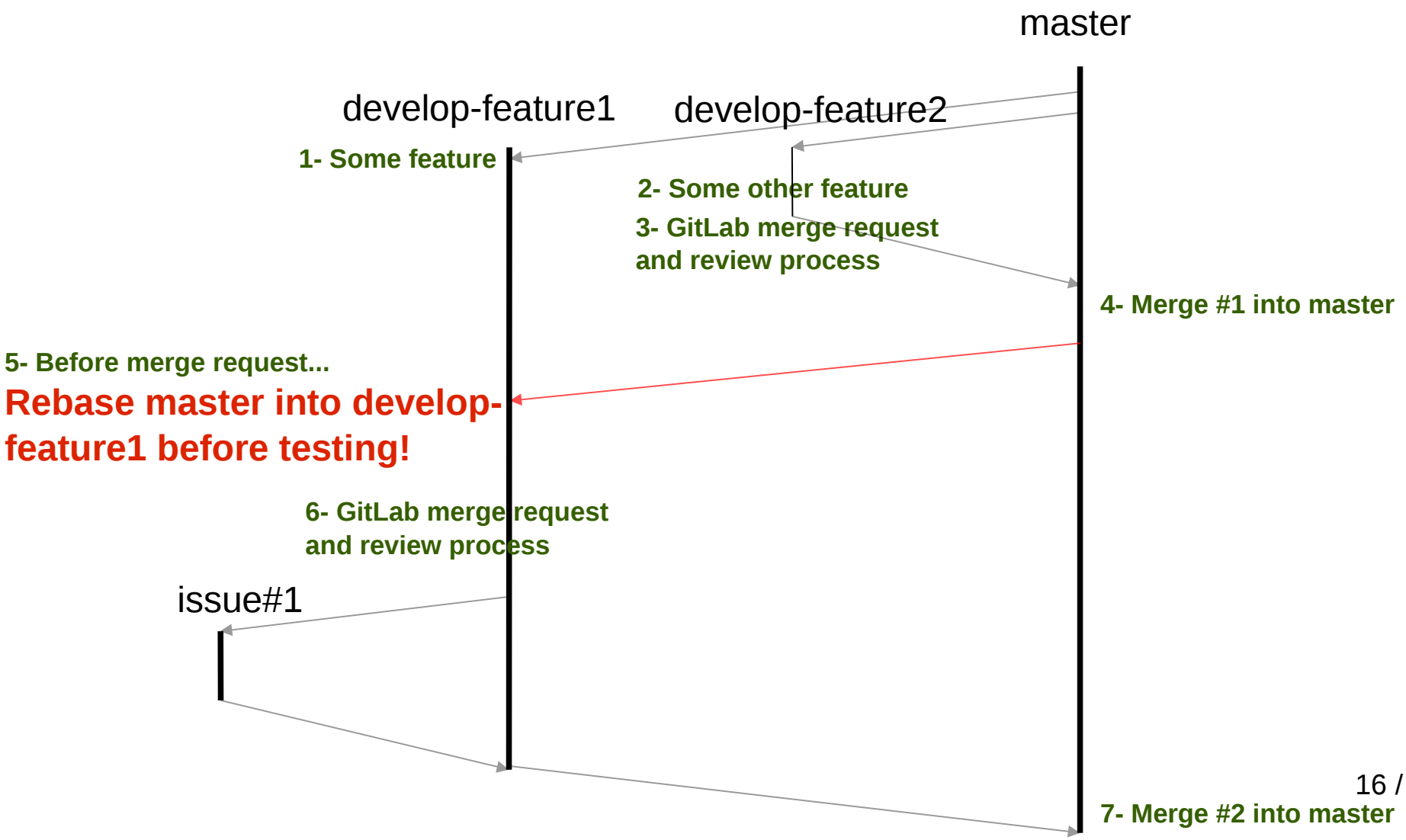
- Move submodule to a particular Tag:

  > *cd submodule_directory*
  > *git checkout v1.0*
  > *cd ..*
  > *git add submodule_directory*
  > *git commit -m "moved submodule to v1.0"*
  > *git push*

- Update submodules: *important: everytime after pull !*

  > *git submodule update --remote --init*
  > *git submodule foreach -q 'branch= *
  *"$(git config -f $toplevel/.gitmodules submodule.$name.branch)"; *
  git checkout $branch'

Develop & Test



To Do

- EC-Earth: v2.3.0a & v3.1c (under ecearth.git) and NEMO: v3.2a & v3.3a (under nemo.git) to be released with respective templates and required submodules.
- Other models ?
- Release EC-Earth test suite (agreed configurations).
- Plugins development.
- Interaction with SVN EC-Earth portal.
- Release spin-up branch and Autosubmit job-wrapper.
- IS-ENES2 – Comparison with Cylc.
- Interface with Cylc.
- Autosubmit paper.